Dynamic Classifier Selection Based on Imprecise Probabilities: a Case Study for the Naive Bayes Classifier

Meizhu Li, Jasper De Bock, and Gert de Cooman

Ghent University, ELIS, SYSTeMS
{meizhu.li,jasper.debock,gert.decooman}@ugent.be

Abstract. Dynamic classifier selection is a classification technique that, for every new instance to be classified, selects and uses the most competent classifier among a set of available ones. In this way, a new classifier is obtained, whose accuracy often outperforms that of the individual classifiers it is based on. We here present a version of this technique where, for a given instance, the competency of a classifier is based on the robustness of its prediction: the extent to which the classifier can be altered without changing its prediction. In order to define and compute this robustness, we adopt methods from the theory of imprecise probabilities. As a proof of concept, we here apply this idea to the simple case of naive Bayes classifiers. Based on our preliminary experiments, we find that the resulting classifier outperforms the individual classifiers it is based on.

1 Introduction

In machine learning and statistics, classification is the problem of predicting the class of a new instance, using only its features and a given data set with labelled instances. A model for performing this task is called a classifier.

The usual approach to solving such problems is to consider some tractable family of possible classifiers and to try to select from among them the classifier that achieves the highest classification accuracy on all instances. The method of Dynamic Classifier Selection [2] switches this around: instead of selecting a single classifier to be used on all instances, it selects a (possibly) different classifier for each and every instance, and then uses the result of this classifier to predict the class of that instance. If all goes well, the resulting combined classifier outperforms each of the individual classifiers it is based on.

The key to the success of this Dynamic Classifier Selection method is to find a way to assess, for a given instance, which classifier is most likely to classify it correctly. We here propose to base this assessment on imprecise-probabilistic measures for the robustness of a prediction: the extent to which a classifier can be altered without changing its prediction. Our main reason for this proposal is the recent discovery that these measures of robustness correlate surprisingly well with the accuracy of their classifier (when evaluated on instances with similar robustness) [3].

To the best of our knowledge, this idea has never been tried before. For that reason, we here start by applying it to the case of the Naive Bayes Classifier, as it is a simple but well-known and surprisingly performant classifier. Furthermore, since the Naive Bayes topology is a special case of the imprecise graphical models that are considered in Reference [3], this will allow us to compute the required robustness measures with existing algorithms.

The rest of this paper is organized as follows. After a brief introduction to the Naive Bayes Classifier in Section 2, Section 3 goes on to introduce its imprecise-probabilistic extension, called the Naive Credal Classifier, and the critical perturbation threshold that can be derived from it. Next, in Section 4, we explain how these thresholds can be used as a tool to choose, for a given instance, between two different classifiers. Experiments on real data sets from the UCI Machine Learning Repository are reported on in Section 5. We conclude the paper in Section 6, where we also hint at possible avenues for future research.

2 Naive Bayes classifiers

A Naive Bayes Classifier (NBC) is a simple probabilistic model that is used to estimate the class of an instance based on the value of its features. The class variable is denoted by C and takes values c in a finite set $\mathcal{C}$. If $\mathcal{C}$ is binary, as will be the case in our experiments, we denote its elements by c and $\overline{c}$. The number of features is denoted by m. For every $i \in \{1, \ldots, m\}$, the i-th feature variable is denoted by F_i and takes values f_i in a finite set $\mathcal{F}_i$. For notational convenience, we gather all feature variables in a single vector $\mathbf{F} = (F_1, \ldots, F_m)$ that takes values $\mathbf{f} = (f_1, \ldots, f_m)$ in $\mathcal{F}_1 \times \cdots \times \mathcal{F}_m$.

For any given feature vector $\mathbf{f}$, a Naive Bayes Classifier will return as its estimate the class $\hat{c}$ that has the highest probability $P(\hat{c}|\mathbf{f})$ given the features $\mathbf{f}$. In fact, it has this in common with every other probabilistic classifier. In the case of a Naive Bayes Classifier, the computation of these probabilities is facilitated by the (naive) assumption that the features are independent given the class; see Fig. 1.

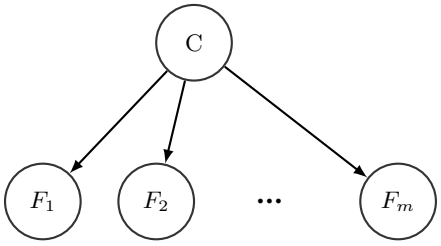

Fig. 1: Example of a Naive Bayes Classifier

Due to this assumption, the conditional probability that is to be maximised is given by

$$P(\hat{c}|\mathbf{f}) = \frac{P(\hat{c}) \prod_{i=1}^{m} P(f_i|\hat{c})}{\sum_{c \in \mathcal{C}} P(c) \prod_{i=1}^{m} P(f_i|c)}.$$

In this expression, the (conditional) probabilities that appear on the right hand side are typically learned from data. To avoid probability zero, we adopt Laplace smoothing, meaning that for all $i \in \{1, \ldots, m\}$, $c \in \mathcal{C}$ and $f_i \in \mathcal{F}_i$:

$$P(c) = \frac{n(c)+1}{n+|\mathcal{C}|} \text{ and } P(f_i|c) = \frac{n(c,f_i)+1}{n(c)+|\mathcal{F}_i|},$$

with n the total number of data points, $n(c)$ the number of data points with class c and $n(c, f_i)$ the number of data points with class c and i-th feature f_i.

Example 1. Most of our experiments further on will be conducted on the Balloons data set of the UCI Machine Learning Repository [4]. It has two possible class values, which we denote by c and $\overline{c}$ and 4 features, which we denote by F_1, F_2, F_3 and F_4. For the purposes of this contribution, the meaning of these features is irrelevant.

The data set consists of 76 instances, which we randomly split in a training set (70%) and a testing set (30%). The training set is used to learn the local probabilities of a Naive Bayes Classifier (using Laplace smoothing), and this classifier is then used to predict the class of the instances in the testing set. This process is repeated a hundred times, and the average accuracy over these hundred runs is reported.

For an NBC that uses all four features, the obtained (average) accuracy is 0.776502. We refer to this classifier as 'Classifier 1'. We also consider a second classifier, called 'Classifier 2', that only uses F_1, F_3 and F_4. For that classifier, the obtained accuracy is 0.746948. After performing feature selection using the Sequential Forward Selection (SFS) method, these two classifiers came out first and second, respectively.

For a given instance, these two classifiers may of course yield different predictions. For didactic purposes and future reference, we here mention one particular unspecified instance where, in the first of our hundred runs, Classifier 1 predicted c while Classifier 2 predicted $\overline{c}$.

3 Naive Credal Classifiers and their thresholds

The Naive Credal Classifier (NCC) [5] is an extension of the Naive Bayes Classifier to the framework of imprecise probabilities that can be used to robustify the inferences of an NBC. Basically, the idea is to consider an NBC whose local probabilities are only partially specified.

In particular, instead of considering a probability mass function $P(C)$ that contains the probabilities $P(c)$ of each of the classes $c \in \mathcal{C}$, an NCC considers a set of such probability mass functions, which we denote by $\mathcal{P}(C)$. Similarly, for every

class $c \in \mathcal{C}$ and every $i \in \{1, \ldots, m\}$, it considers a set $\mathcal{P}(F_i|c)$ of conditional probability mass functions. In general, these local sets can be learned from data, elicited from experts, or obtained by considering neighbourhoods around the local models of an NBC. We here consider the first option. In particular, we use a version of the Imprecise Dirichlet Model (IDM) [1], suitably adapted such that it is guaranteed to contain the result of Laplace smoothing. In particular, $P(C)$ is taken to belong to $\mathcal{P}(C)$ if and only if there is a probability mass function t on $\mathcal{C}$ such that

$$P(c) = \frac{n(c) + 1 + st(c)}{n + |\mathcal{C}| + s} \text{ for all } c \in \mathcal{C},$$

where s is a fixed hyperparameter that determines the degree of imprecision. For every $i \in \{1, \ldots, m\}$ and $c \in \mathcal{C}$, the local set $\mathcal{P}(F_i|c)$ is defined similarly.

If we now choose a single probability mass function $P(C)$ in $\mathcal{P}(C)$ and, for every $c \in \mathcal{C}$ and $i \in \{1, \ldots, m\}$, a single conditional probability mass function $P(F_i|c)$ in $\mathcal{P}(F_i|c)$, we obtain a single NBC. By doing this in every possible way, we obtain a set of NBCs. This set is a Naive Credal Classifier (NCC) [5].

Classification for such an NCC is done by performing classification with each of the NBCs it consists of separately. If all these NBCs agree on which class to return, then the output of the NCC will be that class. If they do not agree, the result of the NCC is indeterminate and consists of a set of possible classes, amongst which it is unable to choose.

Example 2. For the particular instance at the end of Example 1, for the same first run, using $s = 0.6$ makes Classifier 2 indeterminate, in the sense that it then returns the uninformative set $\{c, \overline{c}\}$, while for that value of s, the NCC that corresponds to Classifier 1 continuous to predict class c.

For our present purposes, however, we are not interested in the indeterminate predictions of an NCC. Instead, we are interested in the maximum value of s for which it still remains determinate. This value is a particular case of the critical perturbation threshold in Reference [3]. Quite remarkably, it has been observed that for any given instance, the corresponding critical perturbation threshold serves as a good indicator for the performance of the original NBC: instances with higher thresholds are classified correctly more often [3]. Furthermore, this threshold can be computed efficiently using the algorithms in that same reference.

Example 3. Continuing with Example 2, we find that for that same instance and that same run, the critical perturbation threshold for Classifier 1 is 0.63, while that for Classifier 2 is 0.52. This explains why, for $s = 0.6$, Classifier 1 continued to be informative whereas Classifier 2 was indeterminate.

4 A Dynamic Classifier Selection Method

The main takeway message of the previous section was that, for a given instance and classifier, the corresponding critical perturbation threshold serves as a good

indicator for the performance of that classifier on this instance, in the sense that instances with higher thresholds have a higher chance of being classified correctly. Inspired by this observation, we will now try to use this threshold to perform dynamic classifier selection, that is, to derive a method that, for every new instance to be classified, is able to select the most competent classifier among a set of available ones.

As a first idea, one could consider to simply choose the classifier with the highest threshold. The problem with that approach, however, is that it does not make sense to directly compare the thresholds of different classifiers, because the empirical relation between threshold and performance differs from classifier to classifier. Classifiers with more features, for example, tend to have lower thresholds for all instances, but this does not mean that they achieve lower accuracies. In general, the only thing that we know is that for a given classifier, instances with higher thresholds are classified correctly more often.

What we need, therefore, is a method for determining the empirical relation between the thresholds of different classifiers and their probabilities of correctly classifying the instance that is considered. In particular, for the case of two classifiers that we will here consider, we need a method for determining the empirical relation between a pair of thresholds—one for each of the two classifiers—and the corresponding probabilities of successful classification. Given such a relation, for any given instance, we can use its pair of thresholds to estimate which of the two considered classifiers has the highest probability of successful classification, and then use that classifier to predict the instance's class. We propose to do this in the following way.

For every new test instance that is to be classified, we start by searching the training set for instances that have a similar pair of thresholds. In particular, we choose those k training instances whose pair of thresholds is closest to that of the test instance, according to some given distance measure. In our experiments, we consider two different distance measures: the Euclidean distance and the Chebyshev distance. On the chosen subset of training instances, we then compare the success rate of each of the two classifiers, select the classifier with the highest success rate, and use that classifier to perform the classification. The resulting classifier is a combination of the two classifiers it is based on: for every test instance, it uses the one that is expected to perform best.

Fig. 2 illustrates this idea with a fictitious example that is imagined to have fifty training instances, whose pairs of thresholds are depicted on the plane. The threshold value of Classifier 1 and 2 are the x- and y-coordinate, respectively. Every instance in the training set corresponds to a black point. Consider now a test instance whose pair of thresholds corresponds to the red dot and let $k = 10$. Our method then starts by considering the ten points that are closest to the red dot, according to the chosen distance measure. In Fig. 2, the green diamonds correspond to the Euclidean distance, while the purple stars are for Chebyshev distance. Next, we compare the accuracy of both classifiers on this set of points. Whichever classifier performs best on them is the one that we will use to classify this particular test instance.

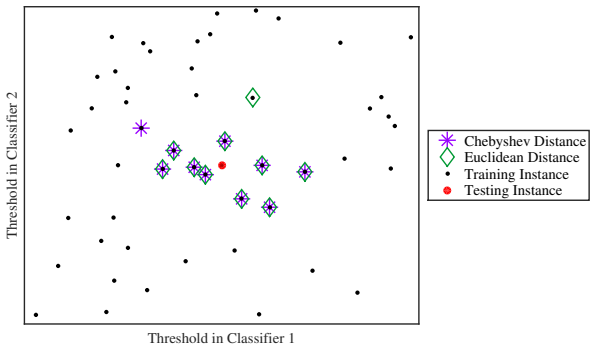

Fig. 2: Illustration of the chosen k-nearest instances, using a fictituous data set with fifty training points, and for $k = 10$ and two different distance measures

Our next example tests this method on the Balloons data set that we considered before.

Example 4. We consider the two classifiers that were discussed in Example 1 and compare them with two new classifiers that are obtained by the method that we introduced above, one for each of the two considered distance measures. As before, we randomly split the data into training and testing instances and report average accuracies over hundred such runs. In order to study the effect of the parameter k, we vary it from 2 to 20. The results are depicted in Fig. 3. The pink line shows the accuracy of the combined classifier with the Chebyshev distance ($\mathrm{AC}_{\mathrm{ch}}$), while the green one depicts the accuracy for the Euclidean distance ($\mathrm{AC}_{\mathrm{eu}}$). The purple line and the orange line correspond to the constant accuracy of Classifier 1 ($\mathrm{AC}_{\mathrm{C1}}$) and Classifier 2 ($\mathrm{AC}_{\mathrm{C2}}$), respectively.

A first important observation is that our new classifiers outperform the individual classifiers on which they are based, regardless of the value of k. We also see that the difference between using the Euclidean or Chebychev distance seems negligible. For $k = 3$, both of our new classifiers reach their peak accuracies, which is given by $\mathrm{AC}_{\mathrm{ch}} = 0.788485$ and $\mathrm{AC}_{\mathrm{eu}} = 0.788961$, respectively. The accuracy of the original classifiers are $\mathrm{AC}_{\mathrm{C1}} = 0.776502$ and $\mathrm{AC}_{\mathrm{C2}} = 0.746948$.

5 Additional Experiments

As a final test, we apply our method to four more data sets, again from the UCI Machine Learning Repository [4]. The Breast Cancer Wisconsin (original) data set (BCW), the Statlog (Australian Credit Approval) data set (ACA), the Auto MPG data set (MPG) and the Tic-Tac-Toe Endgame Data Set (TIC). A brief description of these data sets is given in Table 1, along with the Balloons data set that we considered before. For the sake of simplicity, instances with missing

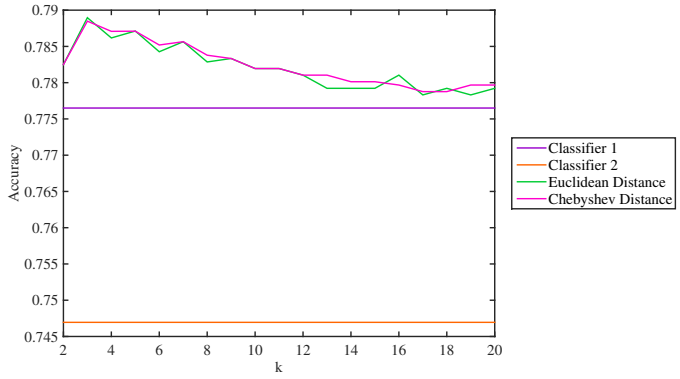

Fig. 3: The achieved accuracy as a function of the parameter k, for four different classifiers: the two original ones (which do not depend on k) and two combined classifiers (one for each of the considered distance measures)

values and features with string values were ignored and continuous variables were discretized by their median. For the remaining features, feature selection was performed to select the best set of features. Here too, the corresponding classifier is called 'Classifier 1' (C1). 'Classifier 2' (C2) is a suboptimal classifier with slightly different features. The features of these two classifiers are reported in Table 1.

Table 1: Description of data sets

Name	# Data	# Class values	# Features	Features C1	Features C2
Balloons	76	2	4	$(1, 2, 3, 4)$	$(1, 3, 4)$
BCW	699	2	9	$(1, 2, 5, 6, 8)$	$(1, 2, 6, 8)$
ACA	690	2	14	$(3, 4, 6, 13)$	$(1, 3, 4, 6, 7)$
MPG	398	2	8	$(1, 5, 6)$	$(1, 4, 5, 6)$
TIC	958	2	9	$(1, 5, 7, 8)$	$(1, 2, 5, 7, 8)$

The results of our experiments are given in Table 2. In contrast with Example 4, the parameter k was not kept constant over all runs. Rather, for each run, an optimal value of k was determined through cross-validation on the training set. Once more, our combined classifiers consistently outperform the individual ones on which they are based. Here too, the choice of distance measure seems to have very little effect.

Table 2: A comparison of all four classifiers

Data Set	AC_{C1}	AC_{C2}	AC_{eu}	AC_{ch}
Balloons	0.776502	0.746948	0.781039	0.781970
BCW	0.974221	0.972496	0.974710	0.974710
ACA	0.723675	0.723190	0.724884	0.724884
MPG	0.920696	0.917610	0.921039	0.920697
TIC	0.724366	0.724363	0.733661	0.732731

6 Conclusions and future work

The main conclusion of this contribution is that imprecise-probabilistic robustness measures, such as the critical perturbation threshold that we here considered, can be used to develop dynamic classifier selection methods that outperform the individual classifiers they select from. Given the restricted scope of our experiments, this conclusion is of course preliminary. We regard our results as a proof of concept, and hope that they will inspire some of you to apply similar techniques to other—perhaps more advanced—types of classifiers as well.

In our own future work, we intend to start by deepening our study of the case of the Naive Bayes Classifier. In particular, we will study the performance of our method on classification problems with more than two classes, and will extend it to allow for dynamic classifier selection among more than two classifiers. Finally, we would like to explore the effect of the specific set of classifiers among which our method chooses. In our current experiments, the classifiers that we started from had similar features and accuracies; it might very well be beneficial to start from classifiers that are more diverse.

7 Acknowledgements

We acknowledge support for this project from the China Scholarship Council. We also thank two anonymous reviewers for their generous constructive feedback.

References

1. Bernard, J.M.: An introduction to the imprecise Dirichlet model for multinomial data. International Journal of Approximate Reasoning, 39(2-3), 123–150 (2005)
2. Cruz, R.M.O., Sabour, R., Cavalcanti, G.D.C.: Dynamic classifier selection: Recent advances and perspectives. Information Fusion, 41, 195–216 (2018)
3. De Bock, J., De Campos, C.P., Antonucci, A.: Global sensitivity analysis for MAP inference in graphical models. Advances in Neural Information Processing Systems 27 (Proceedings of NIPS 2014), 2690–2698. (2014)
4. UCI Machine Learning Repository: http://mlr.cs.umass.edu/ml/index.html.
5. Zaffalon, M.: The naive credal classifier. Journal of statistical planning and inference, 105(1), 5–21 (2002)